GR-FB Block Cleaning Scheme in Flash Memory

A.R. Rahiman[1], N. Manshor[1], A.A. Halin[1], P. Sumari[2], and H. Kamarulhaili[2]
[1] Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 Serdang, Malaysia
Email: {amir, ayu, alfian}@fsktm.upm.edu.my
[2] School of Computer Sciences, Universiti Sains Malaysia, 11800 Penang, Malaysia
Email: {putras, hailiza}@cs.usm.my

***Abstract*—Flash memory is a non-volatitle memory that offers several superior advantages as a data storage device in many electronic gadgets. However its two operational characteristics, namely 1) *Out-place updating* and 2) *Cleaning process* will affect its performance as an efficient storage sub-system if not managed wisely. This paper proposes a new block cleaning scheme, called Greedy and Frequency-based (GR-FB). The scheme combines the victim block selection procedure and the valid-data re-organizing procedure to improve the block cleaning process performance. The trace-driven simulations have shown the GR-FB scheme performs better than existing cleaning process schemes where the cleaning cost is reduced at least by 8% and the wear-leveling level for all blocks in the memory array are nearly equal.**

***Index Terms*—flash memory, cleaning process, wear-leveling, frequency, simulation, data**

I. Introduction

A flash memory is a non-volatile semiconductor storage device that provide faster data access, are small and light weight, has zero noises, boasts solid-state reliability, consume lesser power and very resistant to shock [1 – 4]. Today, it has become the demanding data storage device for storing and transferring data in most electronic gadgets, including MP3 players, personal digital assistants (PDA's), mobile phones, digital cameras, etc. However, its two operational characteristics namely, 1) *Out-place updating* and 2) *Cleaning process* have brought various technical challenges in designing and implementing an efficient storage sub-system. Updating existing data at the same location is strictly prohibited due to the time and the power consuming practice [3]. Therefore, the out-place updating policy has been strongly suggested [3 – 5]. The updated data is stored into a new location, while the original copy is set as *garbage*[1]. When the updating process frequently occurs, the amount of garbage hugely increases resulting in lesser device free space. Thus, the garbage needs to be cleaned to ensure the continuity of the data storing process. In the cleaning process, the block containing large amounts of garbage (hereinafter referred to as victim block) is identified and being cleaned. As the cleaning process in flash memory is carried out on block units, the block may contain valid data that are currently being used by particular applications. Therefore, before initiating cleaning, all valid data residing in the block must be identified and then being copied into available free spaces in other blocks. Moreover, the cleaning process and the block utilization level (the ratio between the valid data size and the block size) substantially impacts flash memory access performance, energy consumption and block endurance [2]. Several studies focusing on identifying the victim block have been proposed in the literature. The objectives of the studies are to minimize the cleaning cost and lengthen the memory block lifespan [1, 4, 6, 11, 13 – 15]. The cleaning cost refers to the total access time required in handling the cleaning process. This paper proposes a new block cleaning scheme, called Greedy and Frequency-based (GR-FB) that combines the victim block selection procedure with the valid data re-organization process. According to the simulation results, the GR-FB scheme perform better than existing block cleaning schemes in terms of cleaning cost and valid data copying amount, as well as guaranteeing memory block lifespan. The remainder of this paper is organized as follows. Section II discusses about flash memory technology and its related works. Then, the GR-FB scheme is presented in Section III. The performance of the GR-FB is shown in Section IV. Finally, conclusions are presented in Section V.

[1] The garbage refers to an obsolete or "*dead*" data and shall not be used any longer.

II. Backgrounds and Related Works

There are two types of flash memory with in the current market, 1) *NOR-flash* and 2) *NAND-flash* [12]. This paper focuses on the NAND-flash and will throughout this paper, refer to it as flash memory.

A. Flash Memory Characteristics

As shown in Fig. 1, flash memory is a blocks-and-pages-based storage device. The page unit is the basic accessing unit and a group of pages form the block unit. Each page unit has two areas, data area and spare area. The data area stores the actual data while the spare area stores assisting information for the data area. Table I shows current flash memory specifications and attributes [16]. Next, data updating in flash memory is performed via the *out-place-policy*. Therefore, the page unit in flash memory can fall into three states namely the *free*, *valid*, and *invalid*. The *free* state contains pages with no data and is ready for data storing. The *valid* state contains pages with the current version of data, whereas the *invalid* state contains pages with garbage. Moreover, the memory blocks containing valid pages are referred to as active block, while blocks containing free or invalid pages are referred to as inactive block [7]. However, each memory block could be tolerant to a limited number of erasure cycles. Excessive erasures will cause the block to become permanently spoiled. For instance, a multi-level cell (MLC) block can support approximately 10,000 erasure cycles. If the same block is erased and then re-programmed every

DOI: 01.IJNS.02.04.512

second, the block would exceed the 10,000 cycle limit in just three hours. To avoid this limitation, a wear-leveling policy that wears down all memory blocks as evenly as possible is necessary [4, 5, 8, 10].

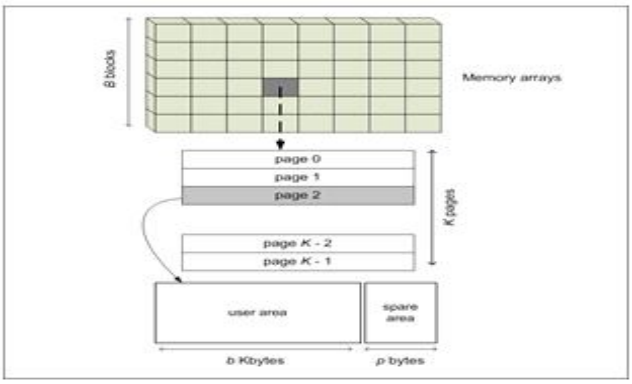

Figure 1. Flash memory blocks and pages layout.

TABLE I. FLASH MEMORY PROPERTIES

Specification	Details
Power consumption	2.70 – 3.60V
Size (h x w x d)	12 mm x 20 mm x 1.2 mm
Access time	
Read (2 KB)	25 µs
Write (2 KB)	200 µs
Erase (128 KB)	1.5 ms
Data endurance	100 K (write and erase cycles)
Data retention	10 years
Noise	~ 0 db

B. Cleaning Process in Flash Memory

The cleaning process is realized in three stages. First, a victim block is determined. Second, all valid data in the block are identified and then copied into free pages in other active or inactive blocks. Third, the victim block is erased upon completion of copying the valid data. Note that, when a particular active block turns into inactive (all pages turn into invalid), the block can be erased automatically without requiring the copying process. A number of studies focusing on determining the victim block to be cleaned have been proposed in [1, 3, 4, 10 – 13]. Two primary goals of these studies are minimizing the cleaning cost and guaranteeing all memory blocks are evenly wear. The Greedy (GR) [1] scheme determines a victim block according to the block utilization level. The block with the lowest utilization level is selected regardless of the block's erasure cycle. Then, Kawaguchi *et al* [3] have proposed the cost-benefit (CB) scheme which selects the victim block that maximizes the function (a x (1–u))/2u). Both parameters u and a refer to the block utilization level and the elapsed time from the last data invalidation. Similar to the GR, the wear-leveling policy is neglected in this scheme. The combination between the victim block selection scheme, called cost age time (CAT), and the valid data re-organizing process, namely dynamic dAta cluster (DAC), has been proposed by [4]. The scheme selects multiple victim blocks according to the lower values from function (u/1-u) x (1/a) x e, where parameter e refers to block erasure counts. In addition, the wear-leveling policy is guaranteed in this scheme. The cost age time with age sort (CATA) [9] scheme improves the CAT scheme by selects multiple victim blocks that have maximum value according to the function (1-u/1+u) x a x (1/e). Also, the scheme guarantees memory block lifespan. Then, the S-Greedy (S-GR) [12] and the Endurant and Fast Greedy (EF-GR) [13] schemes extend the existing GR scheme by considering the wear-leveling policy. The victim blocks are selected according to the block utilization level and both schemes give focus on the valid data re-organization process. The S-GR scheme re-organized the valid data into new blocks according to the Swapped-Out Time (SOT) information while the EF-GR scheme uses the predicted Inter Update (PIU) time. Jang and Han [14] have propose an efficient cleaning scheme by focusing in valid data re-organization. The valid data are classified into three types according to modification frequency- hot data, cold data, and warm data- and different types of data are stored in different blocks. In this scheme, they consider the current frequency of the data when the cleaning process is initiated. From the above studies, we have found that the block cleaning schemes that combine the victim block selection with the valid data re-organizing procedures assists in achieving the cleaning process objectives. Therefore, it has given us motivation to propose a new block cleaning scheme that considers both procedures.

III. GR-FB BLOCK CLEANING SCHEME

The main focus of the GR-FB scheme is re-organizing the valid data in the victim blocks into a new location by using the decrement operation of the current appearance frequency for each valid data.

A. Victim Block Selection Procedure

The GR-FB scheme selects multiple victim blocks according to the block utilization level. The amount of the active blocks is determined according to the predicted I/O arrival rate that is similarly used in the CATA scheme. The predicted I/O arrival rate ($\hat{\lambda}$) is the estimation of the average length for the future arrival of the accessed data (either retrieve or stored). It is estimated by using a time series analysis called autoregressive of order 1 *AR*(1) process [15]. To estimate $\hat{\lambda}$, first, let A_i represent the time series sample of arrival request a_i in the estimated interval length I where the length of the arrival is set to W. Thus, the sequence values of the arrival request can be formed as a time series $\{ a_i^1 .. a_i^m \}$. Then, by using the *AR*(1) process, the number of future arrivals (represented by $\hat{n}_i$) is estimated using the previous sample value. The value of A_i is estimated as the following equation:

$$\hat{a}_i^{j+1} = \overline{a}_i + \rho_i(1) \cdot (a_i^j - \overline{a}_i) + e_i^j \qquad (1)$$

Both ρ_i and $\overline{a}_i$ represent the autocorrelation function and the mean of A_i while e_i^j is a white noise component. In (1), is assumed equal to 0 while is to be estimated values of $\hat{a}_i^j$ for $j \geq H + 1$. Parameter H (History) represents a finite

DOI: 01.IJNS.02.04. 512

sequence of the previously recorded values. Then, as defined by [9], the autocorrelation function ρ_i is calculated using following equation:

$$\rho_i(l) = \frac{E\left[\left(a_i^j - \bar{a}_i\right) \times \left(a_i^{j+1} - \bar{a}_i\right)\right]}{\sigma_{a_i}^2}, 0 \le l \le H-1 \quad (2)$$

Parameter $\sigma_{a_i}^2$ is the standard deviation of A_i while parameter l is the lag between sample values for which the autocorrelation is computed. If the future arrival size $\hat{n}_i$ is set to M intervals, where $M = W/I$, then the arrival requests of $\hat{a}_i^{H+1}, ..., \hat{a}_i^{H+M}$ are calculated using (1). Then, the estimated number of arrivals in W time units is given by $\hat{n}_i = \sum_{j=H+1}^{H+M} \hat{a}_i^j$ and the estimated arrival rate for the future arrival $\hat{\lambda}$ is $\frac{\hat{n}_i}{W}$. By using thevalue, the amount of victim blocks is specified. If the value is high, then, only a single victim block is erased, since erasing multiple blocks will interfere with the flash memory's current I/O operations.

B. Valid Data Re-organization

First, the memory blocks are partitioned into three clusters according to the appearance frequency in the I/O operational behavior, see Fig. 2. The 1st cluster is used to store valid data with irregular appearance. The 2nd cluster stores the average appearance while the 3rd cluster is allocated data with regular appearance. Parameters p, k and s represent the data appearance frequency and the maximum range of frequency for the memory blocks in the 2nd and 3rd cluster.

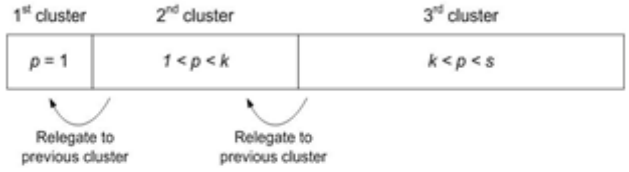

Figure 2. Clustering memory blocks with p frequency appearance

Each valid data in the victim blocks is re-organized into a block in a particular cluster according to their decreased current appearance frequency. For example, if valid data p_i's current frequency is k, then its frequency now becomes $k - 1$. The reason is, since the appearance frequency of the data is solely determined by the user's access behavior, the current appearance frequency may show the current state of the data. There are two conditions for the accessed data in the I/O access pattern. They are either last appearance or will reappear again and again. By decreasing their frequency, the data to be re-allocated in the available blocks in the subsequent cluster can be prevented. Fig. 3 shows the re-organization of the valid data into new blocks according to the decreasing frequency. The valid data with a single appearance are stored in a new block in the first cluster. If the current frequency is 2, the data is demoted from the second cluster to the first cluster. Then, if the decreased frequency is within the same cluster range, the data is re-allocated to an available block belonging to the current cluster. After all the valid data in the victim blocks is copied out into the new blocks, the victim blocks are erased. The decreased frequency is then

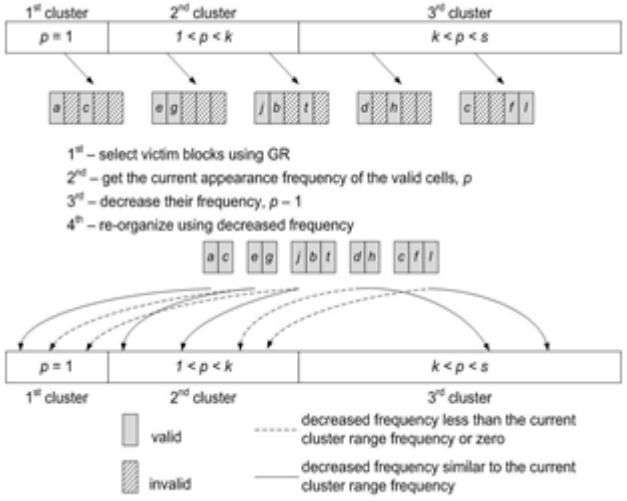

Figure 3. Valid data re-organization in GR-FB scheme.

C. Blocks Wearing Procedure

In the GR-FB scheme, the wear leveling policy is controlled using an efficient valid data re-organization strategy. First, the erasure count for each block in each cluster is recorded. After the block has been erased, its erasure count is increased and the information regarding its erasure count and *ID* are recorded by the data structure called *cluster_list*. Each cluster has an individual *cluster_list* and the block's ID in the *cluster_list* is arranged in ascending order, according to the erasure count. When the data is allocated or re-organized into a particular block, the block with minimum erasure count is selected. When the block has been erased, it will be pushed to the end of the list. Also, if the current active block in the demoted cluster is available with a free page, the re-organization will be stored in the first free page of the block. This re-organization approach ensures all blocks in each cluster is used evenly. The decrement frequency operation helps the valid data to be re-organized into similar blocks each time the cleaning process is initiated.

IV. Simulation Setup and Evaluation

The evaluation was performed using the trace-driven simulation of the real tracing data [16]. The file was collected over a personal computer running general applications such as Web browsers, PowerPoint, Word, and email applications. The details of the file are shown in Table II.

TABLE II. Tracing File Specifications

Time (seconds)	Size (KB)	Total reads	Total writes	Reads /second	Writes /second
365880	14781900	205047	122623	0.56	0.34

By employing the common cleaning process procedure, the GR-FB cleaning process is initiated when the size of the free blocks is fewer than 15% of the total size. The cleaning will

DOI: 01.IJNS.02.04.512

be terminated when the free size exceed 25% of the device's capacity. Three performance paramaters for the evaluations are cleaning cost, valid data copying and block lifespan. The cleaning cost measures the total access time required in handling the cleaning process, while the valid data copying records the amount of data copied or migrated during the cleaning process.

A. Performance Evaluation Results

Fig. 4 illustrates the cleaning cost required for the trace file in the simulation. Compared to exisiting schemes, the cleaning cost required by GR-FB is the lowest. In the figure, the CB scheme consumes the highest cost. By decreasing the current appearance frequency of the valid data, the GR-FB scheme prevents the employment of a new active block during the valid data re-organization process.

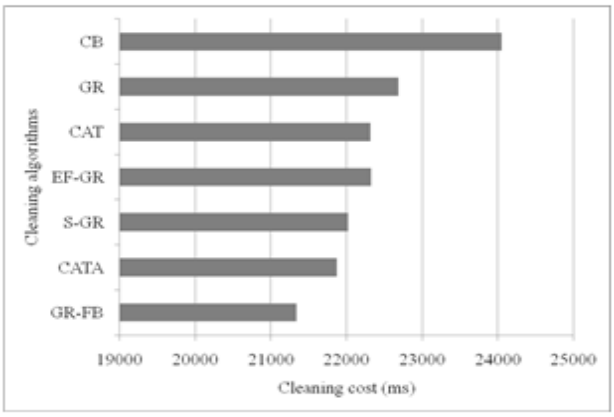

Figure 4. Cleaning cost required for various cleaning schemes.

Fig. 5 shows the percentage amount of blocks involved in the copying process. The smallest percentage value shows minimum block involvement during the copying process, where the copying amount for the GR-FB scheme is not more than 40%. The evaluation on the effectiveness of the wear-leveling policy in the simulation is illustrated in Table III. The performance was evaluated in terms of the average erasure counts for each block in each partitioned cluster by using two levels of frequencies, $k = 2$ and $k = 3$. The similar average value for each cluster indicates that all blocks were evenly accessed during the cleaning process.

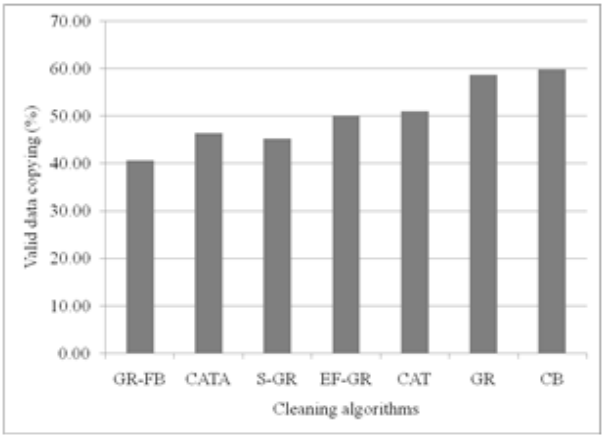

Figure 5. Amount of valid data copying.

TABLE III. PERCENTAGE OF BLOCKS ERASURE COUNTS

Schemes	$k = 2$				$k = 3$			
	C1	C2	C3	Avg	C1	C2	C3	Avg.
CB	59.2	26.9	43.8	43.3	56.6	31.4	48.7	45.6
CAT	56.3	28.7	53.8	46.3	59.4	27.1	50.8	45.8
GR	54.2	26.8	46.4	42.5	47.7	33.6	49.5	43.6
CATA	57.6	31.4	48.7	45.9	56.9	32.8	53.9	47.9
EF-GR	45.8	36.8	46.8	43.1	44.3	37.9	48.3	43.5
S-GR	48.5	37.6	45.7	43.9	48.4	37.8	47.5	44.6
GR-FB	41.8	39.2	37.4	39.5	41.4	39.8	39.3	40.2

* C1 – 1st cluster 1, C2 – 2nd cluster, C3 – 3rd cluster.

The GR-FB scheme records the lowest average value (39.5%) compared to the other schemes. With the different *k* values, the results show significant differences. For example, the average increased from 39.5% to 40.17% when *k* was changed from 2 to 3. From the minimum average values recorded, it is clear that the proposed scheme guarantees memory block lifespan. Unlike the CB and GR schemes, for instance, the erasure count is not the criterion in accessing the block for allocating and re-organization of the access data. The *cluster_list* in each cluster plays a significant role in selecting the block to store the accessed data and the re-organization of the valid data from the selected victim blocks.

V. CONCLUSIONS

The GR-FB scheme emphasizes on combining the victim block selection procedure and the valid data re-organization approach to minimize the cleaning cost and guaranteeing the wear leveling policy. The victim blocks are selected according to block utilization level and the valid data residing in the victim blocks are re-organized into new blocks by decreasing their current appearance frequency. The decrement frequency approach is used because the accessed data is controlled solely by the user's access behavior and it prevents from employing the new block in the next cluster during the re-organization process. The GR-FB scheme performs better than the existing schemes in terms of the cleaning cost and valid data copying percentage. The average percentage of blocks usage in each partitioned cluster is the metric used in evaluating the blocks endurance, and the GR-FB shows that the percentages for all clusters are nearly identical.

REFERENCES

[1] M. Rosenblum and J. K. Ousterhout, "The design and implementation of a log-structured file system," *ACM Transactions on Computer Systems*, vol. 10, pp. 26 – 52, 1992.

[2] F. Douglis, R. Caceres, F. Kaashoek, K. Li, B. Marsh, and J. A. Tauber, "Storage Alternatives for Mobile Computers," *Proc. 1st Symp. Operating Systems Design and Implementation* (OSDI), pp. 25 – 37, 1994.

[3] A. Kawaguchi, S. Nishioka, and H. Motoda, "A flash-memory based file system," *Proceedings of 1995 USENIX Annual Technical Conference*, pp. 155 – 164, 1995.

[4] M.L. Chiang and R.C. Chang, "Cleaning policies in mobile computers using flash memory," *Journal of Systems and Software*, vol. 48, pp. 213 – 231, 1999.

[5] Chang, L.P. and Kuo, T.W, "Efficient management for large-scale flash memory storage systems with resource conservation." *ACM Transactions on Storage*, vol. 1. pp. 381 – 418, 2005.

DOI: 01.IJNS.02.04.512

[6] L.-F. Chou and P. Liu, "Efficient allocation algorithms for flash file systems," *11th International Conference on Parallel and Distribution Systems*, pp. 634 – 641, 2005.
[7] G. Lawton, "Improved Flash Memory Grows in Popularity," *Computer*, vol. 39, pp. 16–18, 2006
[8] L.-P. Chang, "On efficient wear leveling for large-scale flash-memory storage systems," *Proceedings of the 2007 ACM symposium on Applied computing*, pp. 1126 – 1130, 2007.
[9] L.-Z. Han, Y. Ryu, T.-S. Chung, M. Lee, and S. Hong, "An Intelligent Garbage Collection Algorithm for Flash Memory Storages," *ICCSA* 2006, pp. 1019 – 1027, 2006.
[10] NAND vs. NOR flash technology, http://www2.electronicproducts.com/NAND_vs_NOR_flash_technology-article-FEBMSY1-FEB2002.aspx
[11] T.H. Lee, S.H. Park, T.H. Kim, S.G. Lee, and K.D. Chung, "RFFS: Design of a reliable NAND flash filing system for embedded system," *Korea Information Processing Society* (*KIPS) Transactions*, Part A, vol. al2, pp. 571 – 582, 2005.
[12] Kwon, O. Ryu, Y. and Koh, K. "An Efficient Garbage Collection Policy for Flash Memory Based Swap Systems," *ICCSA 2007*, pp. 213 – 223, 2007.
[13] Kwon, O., Lee, J. and Koh, K, "EF-Greedy: a novel garbage collection policy for flash memory based embedded systems," *Proceedings of 7th International Conference on Computational Science (ICCS 2007)*, pp. 913 – 920, 2007.
[14] Jang, K.H. and Han, T.H. "Efficient garbage collection policy and block management method for NAND flash memory," *Proceedings of 2nd International Conference on Mechanical and Electronics Engineering* (ICMEE2010), pp. V1-327 – V1-331, 2010.
[15] A. Chandra, W. Gong, and P. Shenoy, "Dynamic resource allocation for shared data centers using online measurements," *Proceedings of 11th International Workshop Quality of Service (IWQos 2003)*, pp. 381 – 398, 2003.
[16] K9K8G08U1A & K9F4G08U0A_Data Sheet for 512M Bit and 1G Bit NAND Flash Memory SAMSUNG.
[17] A trace of application executions over Windows XP. http://newslab.csie.ntu.edu.tw/~flash/index.php?SelectedItem=Traces

DOI: 01.IJNS.02.04.512